

# EFFICIENT METHODS AND APPARATUS FOR HIGH-THROUGHPUT PROCESSING OF GENE SEQUENCE DATA

This application claims benefit of the priority of U.S. Provisional Application  
Serial No. 60/274,686 filed March 8, 2001.

## BACKGROUND OF THE INVENTION

### 1. Field of the Invention

The present invention relates generally to the processing of gene sequence data with use of a computer, and more particularly to efficient high-throughput processing of gene sequence data to obtain reliable single nucleotide polymorphism (SNP) data and haplotype data.

### 2. Description of the Related Art

Bioinformatics is a field in which genes are analyzed with the use of software. A gene is an ordered sequence of nucleotides that is located at a particular position on a particular chromosome and encodes a specific functional product. A gene could be several thousand nucleotide base pairs long and, although 99% of the sequences are identical between people, forces of nature continuously pressure the DNA to change.

From generation to generation, systematic processes tend to create genetic equilibria while genetic sampling or dispersive forces create genetic diversity. Through these forces, a variant or unusual change can become not so unusual -- it will eventually find some equilibrium frequency in that population. This is a function of natural selection pressures, random genetic drift, and other variables. Over the course of time, this process happens many times and primary groups having a certain polymorphism

(or "harmless" mutation) can give rise to secondary groups that have this polymorphism, and tertiary, and so on. Such a polymorphism may be referred to as a single nucleotide polymorphism or "SNP" (pronounced "snip"). Among individuals of different groups, the gene sequence of several thousand nucleotide base pairs long could be different at 5 or 10 positions, not just one.

Founder effects have had a strong influence on our modern day population structure. Since systematic processes, such as mutation and genetic drift, occur more frequently per generation than dispersive process, such as recombination, the combinations of polymorphisms in the gene sequence are fewer than what one would expect from random distributions of the polymorphic sequence among individuals. That is, gene sequence variants are not random distributions but are rather clustered into "haplotypes," which are strings of polymorphism that describe a multi-component variant of a given gene.

To illustrate, assume there are 10 positions of variation in a gene that is 2000 nucleotide bases long in a certain limited human population. The nucleotide base identifier letters (e.g., G, C, A, and T) can be read and analyzed, and given a "0" for a normal or common letter at the position and a "1" for an abnormal or uncommon letter. If this is done for ten people, for example, the following strings of sequence for the polymorphic positions might be obtained:

Person 1:	1000100000
Person 2:	0000000000
Person 3:	1000100000
Person 4:	1111100000
Person 5:	0000000000

Person 6: 0000000000  
Person 7: 1000100000  
Person 8: 1000100000  
Person 9: 0100000001  
Person 10: 1000100100

5

This list is typical of that which would be found in nature. As shown above, the "1000100000" haplotype is present four times out of ten, the "0000000000" haplotype is present three times out of ten, and the "1000100100" haplotype is present one time out of ten. If this analysis is done for a large enough population, one could define all of the haplotypes in the population. The numbers would be far fewer than that expected from a multinomial probability distribution of allele combinations.

10

The field of bioinformatics has played an important role in the analysis and understanding of genes. The human genome database, for example, has many files of very long sequences that together constitute (at least a rough draft of) the human genome. This database was constructed from five donors and is rich in a horizontal sense from base one to base one billion. Unfortunately, however, little can be learned from this data about how people genetically differ from one another. Although some public or private databases contain gene sequence data from many different donors or even contain certain polymorphism data, these polymorphism data are unreliable. Such polymorphism data may identify SNPs that are not even SNPs at all, which may be due to the initial use of unreliable data and/or the lack of proper qualification of such data.

15

20

In order to discover new SNPs in genes, one must sequence DNA from hundreds of individuals for each of these genes. Typically, a sequence for a given person is about

500 letters long. By comparing the sequences from many different people, DNA base differences can be noticed in about 0.1% - 1.0% of the positions, and these represent candidate SNPs that can be used in screens whose role is to determine the relationship between traits and gene "flavors" in the population. The technical problem inherent to this process of discovery is that more than 1.0% of the letters are different between people in actual experiments because of sequencing artifacts, unreliable data (caused by limitations in the sequencing chemistry, namely that the quality goes down as the sequence gets longer) or software errors.

For example, if the error rate is 3% and 500 people with 500 bases of sequence each are being screened, there are  $(0.03)(500) = 15$  sites of variation within the sequence. If the average frequency of each variant is 5%, and 500 people are being screened, there are  $(0.05)(0.03)(500)(500) = 375$  sequence discrepancies in the data set which represent letters that are potentially different in one person from other people. Finding the "good ones" or true SNPs in these 375 letters is a daunting task because each of them must be visually inspected for quality, or subject to software that measures this quality inefficiently.

Furthermore, one must first amplify regions of the human genome from many different people before comparing the sequences to one another. To amplify these regions, a map of a gene is drawn and addresses around the regions of the gene are isolated so that the parts of the gene can be read. These regions of the gene may be referred to as coding sequences and the addresses around these regions may be referred to as primer sequences. More specifically, a primer is a single-stranded oligonucleotide

that binds, via complementary pairing, to DNA or RNA single-stranded molecules and serves for the priming of polymerases working on both DNA and RNA.

Conventional primer design programs that identify primer sequences have existed for years, but they are not suitable for efficient high-throughput data processing of genomic (very large) sequence data. Some examples of conventional primer design programs are Lasergene available from DNASTar Inc. and GenoMax available from Informax, Inc. Basically, conventional primer design programs pick the best primer pairs within a given sequence and provide many alternates from which the user selects to accomplish a particular objective.

Efficient high-throughput reliable methods are becoming critical for quickly obtaining and analyzing large amounts of genetic information for the development of new treatments and medicines. However, the conventional primer design programs are not equipped for high-throughput processing. For example, they cannot efficiently handle large sequences of data having multiple regions of interest and require a manual separation of larger design tasks into their component tasks. Such a manual method would be very time consuming for multiple regions of interest in one large sequence. The output data from these programs are also insufficient, as they bear a loose association to the actual positions provided with the input sequence. Finally, although it is important to obtain a large amount of data for accurate assessment, it is relatively expensive to perform amplification over several runs for a large number of sequences. In other words, one large amplification is less expensive to run than several smaller ones covering the same genetic region. Because there are constraints on the upper size

limit, several economic and technical variables should be considered when designing such an experiment.

Accordingly, what are needed are methods and apparatus for use in efficient high-throughput processing of gene sequence data for obtaining reliable high-quality  
5 SNP and haplotype data.

### SUMMARY OF THE INVENTION

The present invention relates generally to the processing of gene sequence data with a computer, and more particularly to efficient high-throughput processing of gene  
10 sequence data for obtaining reliable single nucleotide polymorphism (SNP) data and haplotype data. One novel software-based method involves the use of special primer selection rules which operate on lengthy gene sequences, where each sequence has a plurality of coding regions located therein. Such a sequence may have, for example, 100,000 nucleotide bases and 20 identified coding regions.

15 The primer selection rules may include a rule specifying that all primer pairs associated with the plurality of coding regions be obtained for a single predetermined annealing temperature. This rule could allow for the subsequent simultaneous amplification of many sequences in a single amplification run at the same annealing temperature. The rule that provides for this advantageous specification requires that  
20 each primer sequence has a length that falls within one or more limited ranges of acceptable lengths, and that each primer has a similar G+C nucleotide base pair content. The primer selection rules may also include a rule specifying that a single primer pair

be identified for two or more coding regions if they are sufficiently close together. This rule also provides for efficiency as the single primer pair may be used for the amplification of two or more coding sequences. Yet even another rule specifies that no primer sequence be selected for that which exists in prestored gene family data. This rule is important since it avoids identifying primer pairs that may amplify sequences other than those desired.

The method includes the particular acts of reading gene sequence data corresponding to the gene sequence and coding sequence data corresponding to the plurality of coding sequences within the gene sequence; identifying and storing, by following the special primer selection rules, primer pair data within the gene sequence data for one of the coding sequences; repeating the acts of identifying and storing such that primer pair data are obtained for each sequence of the plurality of coding sequences; and simultaneously amplifying the plurality of coding sequences in gene sequences from three or more individuals at the predetermined annealing temperature using the identified pairs of primer sequences.

Reliable single nucleotide polymorphism (SNP) data and haplotype data are subsequently identified with use of these amplified sequences. More particularly, the method includes the additional steps of sequencing the plurality of amplified coding sequences to produce a plurality of nucleotide base identifier strings (which include, for example, nucleotide base identifiers represented by the letters G, A, T, and C); positionally aligning the plurality of nucleotide base identifier strings to produce a

plurality of aligned nucleotide base identifier strings; and performing a comparison amongst aligned nucleotide base identifiers at each nucleotide base position.

At each nucleotide base position where a difference amongst aligned nucleotide base identifiers exists, the method includes the additional steps of reading nucleotide base quality information (for example, phred values) associated with the aligned nucleotide base identifiers where the difference exists; comparing the nucleotide base quality information with predetermined qualification data; visually displaying the nucleotide base quality information for acceptance or rejection; and if the nucleotide base quality information meets the predetermined qualification data and is accepted, providing and storing resulting data (SNP identification data) that identifies where the difference amongst the aligned base identifiers exists.

After providing and storing all of the resulting data that identifies where the differences exist, the method involves the following additional acts. For each aligned nucleotide base identifier at each nucleotide base position where a difference exists, the method involves the acts of comparing the nucleotide base identifier with a prestored nucleotide base identifier to identify whether the nucleotide base identifier is a variant; and providing and storing additional resulting data that identifies whether the nucleotide base identifier is a variant. The providing and storing of such additional resulting data may involve providing and storing a binary value of '0' for those nucleotide base identifiers that are identified as variants and a binary value of '1' for those nucleotide base identifiers that are not. The accumulated additional resulting data identifies is haplotype identification data.



Advantageously, the methods described herein allow for high-throughput processing of gene sequence data that is quick, efficient, and provides for reliable output data.

## BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a block diagram of a computer system which embodies the present invention;

FIG. 2 is an illustration of software components which may embody or be used to implement the present invention; and

FIGs. 3A-3C form a flowchart describing a method of efficient high-throughput processing of gene sequence data.

## DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

FIG. 1 is a block diagram of a computer system 100 which embodies the present invention. Computer system 100 includes a network 102 and computer networks 104 and 106. Network 102 is publicly accessible, and a server 108 and a database 110 which are coupled to network 102 are also publicly accessible. On the other hand, computer networks 104 and 106 are private. Each one of computer networks 104 and 106 include one or more computing devices and databases. For example, computer network 104 includes a computing device 112 and a database 114, and computer network 106 includes a computing device 116 and a database 118. The computing devices may include any suitable computing device, such as a personal computer (PC).

Network 102 may be the Internet, where an Internet Service Provider (ISP) is utilized for access to server 108 and database 110. Database 110 stores public domain gene sequence data. Also, the inventive software is preferably used in connection with and executed on computing device 112 of private network 104. Although a preferred computer system is shown and described in relation to FIG. 1, variations are not only possible, but numerous as one skilled in the art would readily understand. For example, in an alternative embodiment, network 102 may be an Intranet and database 110 a proprietary, private DNA sequence database.

The methods described herein may be embodied and implemented in connection with FIG. 1 using software components 200 shown in FIG. 2. The software may be embedded in or stored on a disk 202 or memory 204, and executable within a computer 206 or a processor 208. Thus, the inventive features may exist in a signal-bearing medium which embodies a program of machine-readable instructions executable by a processing apparatus which perform the methods.

Such software is preferably used in connection with and executed on computing device 112 of private network 104. Preferably, the system functions within the context of a PC network with a central Sun Enterprise server. The program can be loaded and run on any desktop PC that operates using the Linux or Unix operating system. Other versions could also function in a Windows environment. Alternatively, the software could operate on a publicly accessible server and available for use through a public network such as the Internet.

FIGs. 3A-3C form a flowchart which describes a method for efficient high-throughput processing of gene sequence data. This flowchart can be used in connection with software components 200 of FIG. 2 in the systems described in FIG. 1. Beginning at a start block 302 of FIG. 3A, gene sequence data corresponding to a gene sequence and coding sequence data corresponding to a plurality of coding sequences within the gene sequence are read (step 304). Next, primer pair data within the gene sequence data are identified for one of the coding sequences by following a set of primer selection rules (step 306). The set of primer selection rules includes special rules for efficient, high-throughput processing.

For example, the primer selection rules may include a rule specifying that all primer pair data for the plurality of coding regions be obtained for a single predetermined annealing temperature (e.g., 62° Celsius). This rule allows for the subsequent simultaneous amplification of many sequences in a single amplification run at the predetermined annealing temperature. This primer selection rule further specifies that each primer sequence have a length that falls within one or more limited ranges of acceptable lengths. The primer selection rules may also include a rule specifying that a single primer pair be identified for two or more coding regions if they are sufficiently close together, which provides for efficiency as the single primer pair can be used for the amplification of two or more coding sequences. As yet another example, the primer selection rules may include a rule specifying that no primer sequence data be selected for that which exists in prestored gene family data, which is

important since the program avoids selecting primer pairs that amplify sequences other than those intended.

Referring back to FIG. 3A, the primer pair data that were identified in step 306 are stored in association with the coding sequence (step 308), and may be displayed or outputted. If additional coding sequences need to be considered (step 310), the next coding sequence is selected (step 312) and steps 306 and 308 are repeated. Thus, the acts of identifying and storing are repeated such that primer pair data are obtained for each coding sequence within the gene sequence. Once all of the coding sequences have been considered at step 310, the primer sequences are used in the amplification process.

In particular, the plurality of coding sequences in gene sequences from three or more individuals (typically 100s of individuals) are simultaneously amplified in a gene amplification machine at the predetermined annealing temperature using the identified pairs of primer sequences (step 314). In the embodiment described, the predetermined annealing temperature is 62° Celsius, but in practice it may be any suitable temperature. Next, the plurality of amplified coding sequences are sequenced to produce a plurality of nucleotide base identifier strings (step 316). Each nucleotide base identifier string corresponds to a respective sequence of the plurality of amplified coding sequences. In the embodiment described, the nucleotide base identifiers are represented by the letters G, A, T, and C. The partial flowchart of FIG. 3A ends at a connector B 318, which connects with connector B 318 of FIG. 3B.

Single nucleotide polymorphism (SNP) data and haplotype data are subsequently identified with use of these amplified sequences. Beginning at connector

B 318 of FIG. 3B, each string of the plurality of nucleotide base identifier strings is positionally aligned with the other to produce a plurality of aligned nucleotide base identifier strings (step 320). This may be performed with use of conventional Clustal functionality, which is described later below. Next, a comparison amongst aligned  
5 nucleotide base identifiers is performed at a given nucleotide base position (step 322).

If a difference amongst aligned nucleotide base identifiers exists (step 324), nucleotide base quality information associated with the aligned nucleotide base identifiers where the difference exists is read (step 326). This nucleotide base quality information may be, for example, phred values described later below. The nucleotide  
base quality information is then compared with predetermined qualification data (step 328). Next, the nucleotide base quality information is visually displayed for acceptance or rejection by the end-user (step 330). This step is important because phred values in themselves are not entirely adequate for determining quality. The reason is that phred  
uses a relative signal-to-noise ratio, but common sequence artifacts often show as  
signals having high ratios. If the nucleotide base quality information meets the  
predetermined qualification data and is accepted (step 332), resulting data (SNP  
identification data) that identifies where the difference amongst the aligned base  
identifiers exists is provided (step 334). This resulting data is stored (step 336).

If there are additional nucleotide base positions (step 338), the next nucleotide  
base position is considered (step 340) and steps 322-338 are repeated. Thus, steps 322-  
338 continue to execute until all of the differences amongst the aligned nucleotide base  
identifiers are identified. Step 338 is also executed if no difference exists at step 324, if

the nucleotide base quality information is not acceptable at step 332, or if the user rejects the finding based on its visual appearance. The partial flowchart of FIG. 3B ends at a connector C 342, which connects with connector C 342 in FIG. 3C.

After providing and storing all resulting data that identify where differences amongst the aligned nucleotide base identifiers exist, additional acts are performed starting at connector C 342 of FIG. 3C. At a nucleotide base position where a difference exists, the nucleotide base identifier is compared with a prestored nucleotide base identifier in order to identify whether it is a variant (step 344). The prestored nucleotide base identifier is known from the stored data in step 336. This data could be stored as variant nucleotide bases or as encoded sites (for example major, minor).

Next, additional resulting data that identifies whether a given nucleotide base identifier is a variant is provided (step 348). This additional resulting data is stored (step 350) and may be displayed or outputted. Where differences do not exist amongst aligned nucleotide base identifiers, it is assumed that no variants exist. Steps 348-350 may involve providing and storing a binary value of '0' for those nucleotide base identifiers that are identified as variants, and a binary value of '1' for those nucleotide base identifiers that are not. If additional nucleotide base positions need to be considered (step 352), then the next nucleotide base position is selected (step 354) and steps 344-352 are repeated. Step 352 is also executed if no difference is found at step 346. Thus, repeating of the acts occurs so that they are performed for each aligned nucleotide base identifier at each nucleotide base position where a difference exists. The repeating of steps ends when all nucleotide base positions have been considered at

step 352. The combined additional resulting data provide haplotype identification data (step 356).

#### Additional Details Regarding Primer Sequence Selection and Amplification.

Regarding steps 302-314 in FIG. 3A above, which may be referred to as the preamplification process, raw human genome data is used and the method basically draws little maps with the data. Additional details regarding the preamplification process will now be described.

Coding sequences are regions within a gene sequence that encode the protein of a gene. RNA is made from DNA only at these positions. When the RNA is turned into protein, the protein sequence is a translation of the DNA sequence at the coding region. The sequence between coding sequences is called intron, which is a DNA section that divides exons. Exons are the DNA segments that store information about the part of the amino acid sequence of the protein.

The object of the present invention is to survey the coding sequences at each coding region for a given gene in many different people, which is time consuming and expensive using conventional approaches. Therefore, a preamplification strategy is designed so that many sequences can be read in an efficient and inexpensive manner. Amplification uses two addresses, one in front of the region of interest and one behind it. These two addresses define sites where short pieces of DNA bind and are extended by an enzyme called thermus aquaticus (TAQ) polymerase. Preferably, a high fidelity TAQ variant would be used, such as Pfu polymerase. The two pieces of DNA together with the enzyme result in the amplification or geometric increase in the copy number of

the sequence between the two addresses. After amplification, the software processes read and compare many sequences to one another to find out where people differ. Without amplification, there is too little DNA to read.

One object of the preamplification process is to appropriately select these addresses, which are the primer sequences, for each one of the coding regions. Ordinarily, this is not a trivial task. For any given coding region, there are typically large numbers of potential primer pair solutions from which to select, and often most of these would result in an inefficient or failed amplification because of non-specificity. The preamplification process described herein works in connection with a plurality of coding regions for many genes and identifies a plurality of primer regions so that amplification can be performed in a specific, cost-effective, and efficient manner.

The software program accepts as input: (1) a genome database sequence file, which may be many hundreds of thousands of letters long and downloaded from the freely available human genome database (default format for convenience); (2) data (e.g., numbers) that indicate where the coding regions are in the input sequence file. The file containing the coding region data (taken from the annotation of a publicly accessible human genome data file) may be referred to as a "join" file because the data in this file typically resemble the following:

```
join(8982..9313, 1..81, 17131..17389, 20010..20169, 21754..22353)/gene="CES1 AC020766"
```

OR

```
join(81..140,1149..1320,1827..2092,2402..2548,2648..3089)/gene="example gene AC10003"
```



In the second-listed join file above, the first coding region indicated is the region from 81 to 140; the second coding region indicated is from 1149 to 1320, etc. The object is to select a small region of sequence (e.g., 18-22 letters) in front of and behind each coding region in the input sequence file for each coding region identified in the join file.

- 5 These small sequences are the primers and, for each identified coding region, the program finds a flanking pair of primer sequences. These primer sequences are then named and presented to the user.

Using the two input files, the software is designed to more particularly perform the following in association with steps 302-314 of FIG. 3A:

60 (1) Use the numbers in the input join file to identify the coding regions in the input sequence file;

70 (2) Identify or select suitable primer regions around coding regions in the most efficient manner (e.g., sometimes the primers will flank a single coding region, and sometimes they will flank two or even three coding regions if they are close enough to be amplified efficiently);

80 (3) Select primer pairs for the same annealing temperature (i.e., the temperature required to get them to do their job during amplification). Thus, if one designs ten primer pairs all with the same annealing temperature, say 62° Celsius, they can all be used in an amplification machine together as each amplification run uses a single fixed  
20 temperature;

(4) Avoid ambiguous letters (e.g. the letter "n") when selecting primer regions;

(5) Design primers using a strategy to reduce the chance that the primer will be within what is called a "repeat" region. This strategy involves recognizing poly-A stretches, ensuring that the least amount of intron sequence possible is present between the two primers (as repeats tend to be removed from exon boundaries by buffer space);

(6) Display to the user all of the statistics surrounding the selections (as examples, how many letters exist between two primers of a pair, the precise numerical position of each of the selected primers, etc.); and

(7) Output the primer sequences in a database compatible format (e.g., tab delimited) for easy ordering from primer synthesis vendors.

Now the following input join file

```
join (81..140)/ gene="example gene AC10009"
```

and the following input sequence file

```

1 GAATTCTTTC CAGAAGGCTT TCCATTTACT TTTCTAGAT TCATCAGAAG AATCATTATC
61 TACAGCAGCT GTAACGTATT GAAATGTATT TTATGAACAA TAAGACTTGA AAGTTAAAAT
121 TGCTCCTTTA TCCATGTACT GAAGAATAAA TATTGTGAAA GCAGTCATAA AAACAGAAAGT
181 AATCTTTTGG TACCTCTGCA TTAGAACTCT TTATTAACCA GGTGTATTGC CATTCAACAG
241 TAATATTTTG AAAGGAATCT CTATTTTGA GCAGGTTTCA ACTTCTGCTT TTTATTTTAA
301 ACAGTAGACT TGAAATATTC AGTAACCATG CTATAAAGAG CTATGCTGTA AGACAGCTTT
361 TTCTATTTAT AGAGCATGGT TTTGAAATTA TAACAAAGCA TGGGTTTAT CCTGAAATCA
421 TTCATAAATA ACACGTACCA AAACTTAAT ACGGGCTAGC CAGTGTGAGC CAGTGTGACG
```

are considered. For the input sequence file, the number of the first letter of a line is shown at the beginning of each line and there are spaces every ten letters. Typically, there is an annotation before the sequence in the file, such as that shown below, which is ignored by the software:

```
LOCUS      AL355303    157796 bp      DNA                      HTG      08-SEP-2000
```

DEFINITION Homo sapiens chromosome 10 clone RP11-445P17, \*\*\* SEQUENCING IN  
 PROGRESS \*\*\*, 19 unordered pieces.  
 ACCESSION AL355303  
 VERSION AL355303.11 GI:10086110  
 5 KEYWORDS HTG; HTGS\_PHASE1; HTGS\_DRAFT.  
 SOURCE human.

The input join file identifies the coding region, which is underlined in the sequence  
 below:

```

10      1 GAATTCTTTC CAGAAGGCTT TCCATTTACT TTTCTAGAT TCATCAGAAG AATCATTATC
      61 TACAGCAGCT GTAAGTATT GAAATGTATT TTATGAACAA TAAGACTTGA AAGTTAAAAT
     121 TGCTCCTTTA TCCATGTACT GAAGAATAAA TATTGTGAAA GCAGTCATAA AAACAGAAGT
     181 AATCTTTTGG TACCTCTGCA TTAGAACTCT TTATTAACCA GGTGTATTGC CATTCAACAG
     15 241 TAATATTTTG AAAGGAATCT CTATTTTGA GCAGGTTTCA ACTTCTGCTT TTTATTTTAA
     301 ACAGTAGACT TGAAATATTC AGTAACCATG CTATAAAGAG CTATGCTGTA AGACAGCTTT
     361 TTCTATTTAT AGAGCATGGT TTTGAAATTA TAACAAAGCA TGGGTTTAT CCTGAAATCA
     421 TTCATAAATA GCACGTACCA AGACTTGAAC ACGGGCTAGC CAGTGTGAGC CAGTGTGACG
  
```

Short sequences (e.g., between 18-22 letters) in front of and behind this coding  
 region are selected based on a set of primer selection rules. The program then names  
 these two primer sequences and presents them to the user at the end of the analysis.  
 This is done seamlessly for multiple coding regions identified in the input join file.  
 From the example above, the following primer pair data (in small letters) are selected or  
 25 designed for the given coding region:

```

      1 GAATTCTttc cagaaggctt tccatttacT TTTCTAGAT TCATCAGAAG AATCATTATC
     61 TACAGCAGCT GTAAGTATT GAAATGTATT TTATGAACAA TAAGACTTGA AAGTTAAAAT
    121 TGCTCCTTTA TCCATGTACT GAAGAATAAA TATTGTGAAA GCAGTCATAA AAACAGAAGT
    181 AATCTTTTGG TACCTCTGCA TTAGAACTCT TTATTAACCA GGTGTATTGC CATTCAACAG
    241 TAATATTTTG AAAGGAATCT CTATTTTGA GCAGGTTTCA ACTTCTGCTT TTTATTTTAA
    301 ACAGTAGACT TGAAATATTC AGTAACCATG CTATAAAGAG CTATGCTGTA AGACAGCTTT
    361 TTCTATTTAT AGAGCATGGT TTTGAAATTA TAACAAAGCA TGGGTTTAT CCTGAAATCA
    421 TTCATAAATa gcacgtacca agacttgaac ACGGGCTAGC CAGTGTGAGC CAGTGTGACG
  
```

Since there are typically about ten important regions in a given sequence, there are typically about twenty short primer sequences which are produced. Oftentimes, however, a single primer pair that flanks two (or more) coding regions is picked so that the actual total number of identified primer pairs will be less than two times the number of coding regions.

The two sequences are also named according to specific rules. Here, the names for the example as TPMTE2-5 and TPMTE2-3 are given. The two primer sequences are presented to the user in the output form below.

```
TPMTE2-5 ttccagaaggctttccatttac
TPMTE2-3 gttcaagtcttggtacgtgct
```

Note that the TPMTE2-5 sequence is identical to the first picked sequence whereas the second sequence, TPMTE2-3, is the reverse and compliment of the second picked sequence.

In the preferred embodiment, the following set of primer selection rules are used for selecting primer sequences:

Rule 1: The number of combined "G"s and "C"s should be roughly equal the number of combined "A"s and "T"s.

Rule 2: There should be no longer than four consecutive "G"s together (e.g., ...GGGG...), four consecutive "C"s together, four consecutive "A"s together, and four consecutive "T"s together.

Rule 3: The length of each primer sequence should fall within the range of 18-22 (inclusive). The length is determined by giving a value of four for each "G", four for each "C", two for an "A", and two for a "T", and then calculating the sum of numbers such that the total sum for any selected sequence must equal 62. Thus, depending on the number of "G"s, "C"s, "T"s and "A"s, the total length of sequence necessary to get a value of 62 will

usually fall within the range of 18 to 22 letters (inclusive).

Rule 4: The number of letters that fall in between the two selected sequences (herein referred to as a "block") should be equal to some rough integer multiple of 420 letters. For example, the number can be 420, 840, 1280, 1700, or 2120 (2120 is the maximum and 420 is the minimum). The number of letters does not need to be exactly 420, 840, or 1280, etc. however, but can be reasonably close; say plus or minus 50 or even 75. This range also can be chosen arbitrarily at first and then modified later. For example, if plus or minus 50 is chosen, the range should be 370-470, 790-890, or 1230-1330, etc.

Rule 5: At least one of the primer sequences must be within 100 letters of the beginning or the end of the coding region.

Rule 6: If the size of a block is larger than 1400, a third short sequence should be picked to reside roughly at position "700" in that block. This sequence should have the letters "seq" at the end of its name. For example, in the sequence below, the block is 2290 letters long:

1 GAATTCTttc cagaaggctt tccattttacT TTTCTAGAT TCATCAGAAG AATCATTATC  
61 TACAGCAGCT GTAACGTATT GAAATGTATT TTATGAACAA TAAGACTTGA AAGTTAAAT  
121 TGCTCCTTTA TCCATGTACT GAAGAATAAA TATTGTGAAA GCAGTCATAA AAACAGAAGT  
181 AATCTTTTGG TACCTCTGCA TTAGAACTCT TTATTAACCA GGTGTATTGC CATTCAACAG  
241 TAATATTTTG AAAGGAATCT CTATTTTGA GCAGGTTTCA ACTTCTGCTT TTTATTTTAA  
301 ACAGTAGACT TGAAATATTC AGTAACCATG CTATAAAGAG CTATGCTGTA AGACAGCTTT  
361 TTCTATTTAT AGAGCATGGT TTTGAAATTA TAACAAAGCA TGGGTTTAT CCTGAAATCA  
421 TGCTCCTTTA TCCATGTACT GAAGAATAAA TATTGTGAAA GCAGTCATAA AAACAGAAGT  
481 AATCTTTTgg tacctctgca ttagaactcT TTATTAACCA GGTGTATTGC CATTCAACAG  
541 TAATATTTTG AAAGGAATCT CTATTTTGA GCAGGTTTCA ACTTCTGCTT TTTATTTTAA  
601 ACAGTAGACT TGAAATATTC AGTAACCATG CTATAAAGAG CTATGCTGTA AGACAGCTTT  
661 TTCTATTTAT AGAGCATGGT TTTGAAATTA TAACAAAGCA TGGGTTTAT CCTGAAATCA  
721 TGctcctttg tccatgtact gaagAATAAA TATTGTGAAA GCAGTCATAA AAACAGAAGT  
...1000 bases ...  
1781 AATCTTTTGG TACCTCTGCA TTAGAACTCT TTATTAACCA GGTGTATTGC CATTCAACAG  
1841 TAATATTTTG AAAGGAATCT CTATTTTGA GCAGGTTTCA ACTTCTGCTT TTTATTTTAA  
1901 ACAGTAGACT TGAAATATTC AGTAACCATG CTATAAAGAG CTATGCTGTA AGACAGCTTT  
1961 TTCTATTTAT AGAGCATGGT TTTGAAATTA TAACAAAGCA TGGGTTTAT CCTGAAATCA  
2021 TGCTCCTTTA TCCATGTACT GAAGAATAAA TATTGTGAAA GCAGTCATAA AAACAGAAGT  
2081 AATCTTTTGG TACCTCTGCA TTAGAACTCT TTATTAACCA GGTGTATTGC CATTCAACAG  
2141 TAATATTTTG AAAGGAATCT CTATTTTGA GCAGGTTTCA ACTTCTGCTT TTTATTTTAA  
2201 ACAGTAGACT TGAAATATTC AGTAACCATG CTATAAAGAG CTATGCTGTA AGACAGCTTT  
2261 TTCATAAATa gcacgtacca agacttgaaC

At the region around the letter at position "700", one cannot find a third short sequence that meets the criteria of having roughly equal G+C and A+T. A suitable sequence

around position "723", however, can be found and is shown in lower case. In this example, three sequences are presented to the user: the first two read exactly as they appear in the lower case letters, and the last one being a reverse and compliment of the sequence at position "2270":

TPMTE2-5 ttccagaaggctttccatttac  
 TPMTE2-seq ggtacctctgcattagaactc  
 TPMTE2-3 gttcaagtcttggtacgtgct

The following is a logic summary for the primer identification rules according to the preferred embodiment:

(1) Define the smallest block of sequence that surrounds and completely encompasses the coding region and is either 700 (+/-100) letters long, 1400 (+/- 100) letters long, 2100 (+/-100) letters long, 2800 letters long (+/-200). That is, identify the smallest such block from those having a length =  $n \cdot (700 \pm 100)$  for  $n = \{1, 2, 3, 4\}$ .

(2) Find a sequence at the beginning of this block such that:

(a) the sequence is 18-22 letters long;

(b) the value of the sum of the letters is exactly 62, where a G=4, C=4, A=2 and T=2. Put another way,  $\text{Sum (T)} \cdot 2 + \text{Sum (A)} \cdot 2 + \text{Sum (G)} \cdot 4 + \text{Sum (C)} \cdot 4 = 62$ . Using this rule, G+C will be either 9, 10, or 11 since only with these values is it possible to have a sequence that is 18-22 letters long with the sum of values = 64;

(c) No greater than four of the same consecutive letters must exist (e.g., ...TTT... is fine but ...GGGG... is not) and, if a string of four letters exist in the "5" prime primer, the same string of four or three letters should not exist in the "3" prime primer; and

(d) the last letter should be a "G" or a "C", not an "A" or a "T".

(3) Find a sequence following the end of the block such that the sequence follows the same rules as described in (2) above.

(4) After identifying two or more blocks, if two blocks can be constructed in the input sequence such that the end of one block overlaps with the beginning of another, or such that the end of one is within, say 100 letters of the beginning of another, the two blocks are merged, as long as the new merged block is not greater than 2800 (+/-200). It is preferable to have one large block compared to two or more smaller ones. If the blocks are merged, the first sequence selected for the first block and the last sequence selected for the second block forms the two sequences of the new merged block. The second sequence for the first block and the first sequence of the second block are discarded.

The selected sequences are also named by the software, preferably as follows.

There are three parts to the name. The first is the gene which is the same as the input sequence file name. For example, for the gene "TPMT" all sequences the program finds for the input sequence file will have "TPMT" in the name. In addition, the first block found includes in its name "E1", the second block found includes in its name "E2", the third "E3", and so on. If two blocks are merged, however, both of these tags will be included in the name of the merged block in order. For example, if "E1" and "E2" blocks are merged, then the characters "E1E2" will be in the new name for the new merged block. Finally, the first sequence found for a block will have the characters "-5" and the second will have the characters "-3".

Below is a naming example where there are five blocks and two sequences for each block, except where blocks "2" and "3" were merged, and the merged block is 1260 (+/-100) letters long and required a third sequence to be selected:

TPMTE1-5  
TPMTE1-3  
  
TPMTE2E3-5  
TPMTE2E3-3

TPMTE2E3SEQ

TPMTE4-5

TPMTE4-3

5

TPMTE5-5

TPMTE5-3

Another way to describe the naming process is presented. The 5-prime and the

10 3-prime primer may be presented to the user based on the following logic:

(1) The name of the gene (which is the sequence file name) and block appears in the name of each primer sequence;

15

(2) The gene and block name corresponding to the sequence file is provided in front of the name for a block is provided. If the sequence file is named "AHR", for example, the first block name would include "AHRE1" and the second block name would include "AHRE2";

20

(3) The "5" prime or "3" prime designation is also presented in the name of the primer. For example, the primers for the first block of the AHR gene would read:

25

AHRE1-5 - the first sequence found (sequence whose numerical position is least - e.g. at position 60)

30

AHRE1-3 - the second sequence found (sequence whose numerical position is most - e.g. at position 420)

35

After naming, the sequence of letters for each primer sequence may be presented

as follows:

35

1. Present the first sequence (called the "5" primer) as it appears in the sequence, letter for letter but without the blank spaces;

2. Present the second sequence (called the "3" primer) such that

40

a. The sequence is reversed such that the end is now the beginning and the beginning is now the end and then,

b. "A" is substituted for each "T"

c. "T" is substituted for each "A"

d. "G" is substituted for each "C"

45

e. "C" is substituted for each "G"

(For example: "AATTATGCCT" would become "AGGCATAATT")



3. Present any third sequence for a block (if necessary because the block is 1260 +/- 100 letters long) as it appears in the input sequence exactly, letter for letter but without blank spaces.

An example output looks like:

```
TYRE15 TTGCATGTTGCAAATGATGTCC
TYRE13 CAACCCAGGTCATCGTTCAC
```

```
TYRE25 CCTCTCAAGCACATTGATCAC
TYRE23 TATACTGATCTGAGCTGAGGC
```

and so on, until...

```
TYRE9-5 TAACATTACACTAATGGCAGC
TYRE9-3 TGCTTCTCCTCTAGAGGCTG
```

The numerical position of each primer sequence relative to the input sequence is preferably presented as well.

The following is an example summary of a join file, a gene sequence file (including relevant portions only for brevity), and output data, for the gene "CES1 AC020766". In the gene sequence file below, the coding regions are highlighted in bold print.

```
=====
JOIN FILE FOR GENE "CES1 AC020766"
```

```
join(80513..81472,81911..82007,82114..82219,85116..85265,89595..89651)/gene="
CES1 AC020766"
=====
```

```
=====
GENE SEQUENCE FILE FOR "CES1 AC020766"
```

```
1 aacttagcaa acacatgata ttgtatatag tagacatcat tattgttttc ccctctattc
61 ttcttttcaa tttctgaatc ataaggattg cctgagccta ggagatcaag gccagccttg
121 gcaacatggc gaaatgccat ctctacaaaa aaaaaaaaaa aaattatcta ggtgtggtgg
181 caagcaccag tgggccagc tactcagaag gctgaggtgg gaggattgct tgagcccagg
```

```
*
*
*
```



55

84301	taacggggcca	tttttcatct	ttgtgaatat	tcttggataa	tggtatcagc	agtgctagat
84361	cttaggttcc	ccagacgtat	aacaaaggag	tgcttttgtt	cggctttttg	gcaagatgat
84421	tgcaaaaaag	gtaataaact	ctcactctta	tttttccctt	catttghtaat	gatctaattt
84481	acacagtact	caatatttgg	gaaattctaa	tctccccaac	gtgaggaagt	ggttgaggat
84541	tagcaaagca	ataagtgttt	agcaaattgc	taatatagta	caagtgaaga	acttcagaat
84601	ctgcttgaat	tctgttaaat	gcagcaacta	aataaatgcc	acctcaccat	tttggatgca
84661	gtagtgatta	ttcctccaaa	gcatccagct	aacaaatgaa	ctttattccc	tggggcacac
84721	agatccagtt	tghtaatttac	agatatctca	ccttccatgg	agaattcaca	tcagtagaaa
84781	ttatattaag	aatacctcac	agctgcaaat	acaaagctgc	agctttactt	agaatgttat
84841	ttgcattaaa	aaatcaattt	ttatagctct	aagattctag	agaagctata	ttctatttaa
84901	tacacataaa	caatacaaaa	atgatagtaa	aagtttaaaa	cttagacatc	tgttttttaa
84961	ataaattaaa	gtttttaaac	acgcataaaa	attcatcgca	ctgaaaaaag	gaagcaaaca
85021	gctttaaagg	agtagttggg	taaaaacata	ttaaaaaacc	acgcaagtct	ccaaggaaca
85081	aagtttgact	tttgtaaaac	agtggaaaat	tttaccttaa	ttttatcaat	gtaattcact
85141	tctctgtgat	tgaacacttc	atgggctcca	ttttgcaaaa	caatcttttg	tccttctcca
85201	gtaccagcag	tgcccaaaat	ctttaagcca	taagctctag	caatttggga	tgctgctaatt
85261	ccaacctgaa	aaacaaatat	aacccaagag	ttatatattc	tctacactcc	tgtaaacact
85321	taaatacata	caatgaactt	aagatttcta	taggaccac	cctaacttta	aggaacttaa
85381	gagtgtaaat	gaagaaataa	gaaaaacagc	taactttaat	tgagcattta	aaatattcca
85441	ggaaccatac	taataataatt	ctacatatgt	ttttattcta	tcctcacaat	gaccctataa
85501	agtagtact	tattattgtcc	ctattgtaca	gataagaaa	ttgaagcttc	aaattataag
85561	taatttggcc	aagtcatatg	cggagatgga	aacaggagtt	agaccagtct	gactgcagaa
85621	cttgagtttt	taaccactgc	atcaagatgt	ttgcagggtt	taaagatgat	cagaacatgc
85681	tctctgactt	ctttgtgcat	atgaaattct	aaataacaaa	tghtaaggcct	ccaccattta
85741	agtagaagag	ataggtatat	gggcaaatta	actaattcat	ccatatgggtg	aatgtttata
85801	gagtgtttac	gatgtgctag	acatgggtact	taatghtaaga	aataaaactta	tattctaagg
85861	gtggaggaag	ataatagtca	tatgaatgaa	taaaataaat	tcaggaaata	aaagtgtctaa
85921	gaaaaaataa	gactggctgt	tgggttaaag	agacaggaat	aggggctatt	taggtcatca
85981	ggaagagcca	ctctgaaaaa	atgagacctg	aaaaaagtga	ggaacaagcc	acgagaacat
86041	ccggtcagcc	acgtggagga	tgctgtgggc	atagtgaatg	gccatggcta	acctggcgag
86101	gtgggaatgc	agttgggggc	aaagaacaga	aagaggggca	gtgtgtctca	gggagggggc
86161	tgtacgaaag	ggtcgaagat	gaggccagaa	aggccaagtc	acacagaatc	tgaggggtga
86221	gggtagaggc	ttccgagtat	attaaaacct	gtgcagaacc	acgggagagc	ttaagccagg
86281	aaatgatctg	gttgactcag	gctttaaaaa	ggttgtctca	attacatgtg	aggcacaaag
86341	aaagcggtga	ggaaaatggg	aggaggaaga	tcagtttghta	gctgttagaa	cagctctagat
86401	aagagatgaa	gctggcttga	acaaaggtgg	tggcactgga	aaaaataaac	aaattcagat
86461	atagtttaga	ggtaagctaa	tgggacttcc	tcacagattg	aatgcgggag	atgaggaaaa
86521	gagaaaaata	caggctgtct	cctatgtctt	tggccagatt	aactgggtag	agtgagaaga
86581	ctggagaaca	ctaagtttgt	gaaaatctcc	agatttccact	ttgccaaagt	tgggtggcgca
86641	tgcctgtaat	cccagctatg	tgggaggctg	aggcaggagg	atcgcttggg	cccaggaatt



148081 ttgtctctct taattacatg gagaatgata tagtgactcc ttcatgcctt tttttctcct  
 148141 taacaagcca tatgcaggaa agtttccatg ctgcgcaaac ataaaagaaa gttatatctc  
 148201 attcctaana gaaaactgaa aagc

5 =====

=====

OUTPUT FROM PROGRAM

NUMBER OF JOINS 4

10

1. 80513.... 81472  
 2. 81911.... 82219  
 3. 85116.... 85265  
 4. 89595.... 89651

15

JOIN NUMBER ----- 1  
 Length of pair 959  
 Starting position of block 79813  
 Block length (700 + pairlength +800) 2459

20

Block ...  
 agtttggcctagttttggccttcaaaattaattaaactgtatcaatgtattttgaagtgttaagtcacatctgtatgcttt  
 agctccttctatagatgaggcaaatatacaaacagattaaactgacttttacagaataattattcttttaccttggt  
 tacatggaaaggaatcctccatttttaggatgcacataaaatgccagcctatgttgatgacattgccttaacactttt  
 tttttaagtaattttacagggtagtttaacctgtaaaagaaacagtgagataaacttgaaaatgctaatagcacaaaac  
 acttcagccatggcacatacaaccagaagccaatgatatccttcaactatagaaattagcgggtgtttctgtttatt  
 cctgaagcaggattccatattcaagccagaaattgtcattcaacagaaaaaatcagggtcaaaacaatcaatcacata  
 atgtagcaagacaaaagtatgtgcttatgtgaagaaaaacaaaaacaaaataaccgaacttttattttcttgaat  
 ataattattgatggcaagattgctaagagggtcatccctgtatttttagtttagataaaggcttccagcatagaacactgt  
 taagaagttaactgtcaggagctatgcagaagtgtgagaggcaaaataatataaaaactagaaaagcagggttttaatt  
 ttctatagactttattacacattattatgttacgagacaaatgcagataattcttaattttatcaaatttgtgagctt  
 aattaacaaaaatatttgaccctcaccagaaaaacagataactctaaatctactctgaaaatctaataattgcgaa  
 gtattacctattttggagactatgtattatatcaaagataaagctactattctcacagaacatatggggtcattggca  
 gccaccaataatgaagtaaatattctaattttgggaaaatactgagaaaactaataaattgtcctggatattatt  
 tattcttgccctttacaaaagacttacacatccaaatgagattagtttagaataagagggttttagttcagaaaatgtt  
 caaagtccaatacagtcagtggaatcagagactagagaacctttataaaggtaagtaggcttgaaaacccttgga  
 actgagcagtccttattttgaactagcatgttttaatacaagggtatggaattaatacaaatcaattaagaattactg  
 gaatgcacactcatgccaaatgacaactaacatgttatttctactatgatgactctttgatttgagtgcagatggca  
 taaaaaaatattgctagctatacaataaaattttactcttctgctctctctaaagaaaaatcttattttttcac  
 ataagaagctcatggaatcgaatgttaattaaagaaaagatagggttaagtacaactgggggaaagacagtacctcta  
 attacataggaaatccatgaaagaattaatcatcataagagaagaatcatttttccagtagccccactaccatgaat  
 gatattttcatgagcctcggccaccttctccaatggatattgagaacctatcacagggttcaaccagccaatttcca  
 ttccagcttgaagggtgctgcatattgctgaaattcctcctaagaaaaggaaaaacaaatttctttttgtagtga  
 ccgatgatttaattttcagaagcattaaaaacacttcagaatctaagtgttataccatgaagagtctcttacaat  
 gtgtgacttttgcacttgcagaaactatagaaaaagtagttatctacagggttaaccataaatcccatctgcctg  
 agacagtgttagtgtacaaaatacctgttgctcctgaaattattactagtatcacatttctatctcaaaagggtatgct  
 tacctggatataaattatactgtcacccctagtgtgcttctggtgactaatccttaccactccactagtcatata  
 actaagtttaacatctattcaaaccttcagcttgctgagtaggcaaaactgtaccaatgtttaagttacaaaaatca  
 gaagtacttcttttctaccttggttgaggaaaagagagtaactccaattatactcgactcctttgccatggtgtct  
 cgtgggtttatttcaatagtacctctgctgccaacaacctaactgaaaaacagcaattctacagttaaagattact  
 gtaaaatagtgttaaatgttggtaaaacattaaagtggtaaaaaaaaaaaaaagaaaaggaaacttactatcactc  
 gtccctcatgtgacagaagactcaagtccttactaagatttacattagctaactttcaataattatatcaattcct  
 ttctcaccaacatacttctatataataaaaagagaaatgtagagtaagtagcaagtgaaaaactgtaaaatag□

55 Actual comp position 80450 sequence tatgcagaagtgtgagagggc

Reverse comp position 80450 sequence gcctctcatcacttctgcata  
g c t a toalno totalvalue 8 2 4 7 21 62

5 Actual comp position 81019 sequence tactggaatgcacactcatgc  
Reverse comp position 81019 sequence gcatgagtgtgcattccagta  
g c t a toalno totalvalue 4 6 5 6 21 62

10 JOIN NUMBER ----- 2  
Length of pair 308  
Starting position of block 81211  
Block length (700 + pairlength +800) 1808

15 Block ...  
tggaatcgaatgttaattaaagaaaagatagggtaagtacaactgggggaaagacagtagcctctaattacataggaa  
atccatgaaagaattaatcatcataagagaagaatcatttttccagtagccccactaccatgaatgatattttcatg  
agcctcggccaccttctccaatggatattgagaacctatcacagggtttcaaccagccaatttccattccagcttgaa  
gggctgctgcatattgctgaaattcctcctaagaaaaggaaaaacaaatttctttttgtagtgaaccgtatgattta  
20 attttcagaagcattaaaaacacttcagaatctaagtggtataccatgaagagtctcttacaatgtgtgacttttg  
tcaacttgccagaactatagaaaaagtagttatctacagggttaaccataaatcccattctgcctgagacagtggttag  
tgtacaaaatacctgttgctcctgaaattattactagtatcacatttctatctcaaaagggtatgcttacctggatata  
aattatactgtcaccctagttgtccttctgggtgactaatccttaccaactcccactagtcataataactaagtttaac  
atctattcaaactttcagcttgccctgagtaggcaaactgtaccaatgtttaagttacccaaatcagaagtagtcttct  
25 ttcctaccttggttgaggaaaagagagtaactccaattatactcgactcctttgccatggtgtctcgtgggtttatt  
tcaatagtacctctgctgccacaacctaacatgaaaaacagcaattctacagttaaagattactgtaaaaatagtgt  
taaatgtggtgtaaaacattaaagtggttaaaaaaaaaaaaaagaaaaggaatacttactatcactcgtcctccatgtg  
acagaagactcaagtccttactaagatttacattagctaacatttcaataattatatcaattcctttctcaccaaca  
tacttctatataataaaagagaaatgtagagtaagatagcaagtgaaaaactgtaaaatagctactatctgtacaag  
30 atattatagaaatatgtttcaaagatataataatgctacatctttgagactaataatgcaaaattttaataatct  
aattatataatcacgatgtaattccaaggtaccagccagaacatctaaactgataaaaaatttgactaaatacattg  
ctgtagtgaataaaagtttgtctggaattttcaggtgctagactcaacttgagtataaaatacttagctgaaaattt  
tctatctgtaaaaataaaactttcataaaagaaacaataaatcaaaagccccaaacccccagggggtcccatttttatt  
aataaaacaaaaagcaaaagaagatatcattagctgttcggttttgcatgatttttggtgttttagtgcatgttggtt  
35 tgttctaaatggtttatcatctgtttgatgcactaactctttgggctcttggtgttggtgacgctggctcttcaaaa  
aagctacacacatctacattatattcattttattttaacacacacacacaaatgaatccctgtgcccgggattgcac  
taggtaccaggaatacaaaatacaaacatagggagctcaaaacaaaactagtggagaaagatgggaaatactacagtca  
tagctataaaagtaatgggctaagtaaacacattagcagaaataaatcatagaatacagagaaaaaagggttaagggttg  
attgcctgccatggtcagataaaagttccacagagacga

40 Actual comp position 81844 sequence gcttgcctgagtaggcaaac  
Reverse comp position 81844 sequence gtttgcctactcaggcaagc  
g c t a toalno totalvalue 6 5 4 5 20 62

45 Actual comp position 82362 sequence tgtaattccaaggtaccagcc  
Reverse comp position 82362 sequence ggctggtaccttgggaattaca  
g c t a toalno totalvalue 4 6 5 6 21 62

50  
JOIN NUMBER ----- 3  
Length of pair 149  
Starting position of block 84416  
55 Block length (700 + pairlength +800) 1649  
Block ...



ttttcttttttactttccagttcacatgactacttttctagtatgtactgaaaagaagggacatgcagcaaggcatgag  
 gggatgcctcactattccagatggacggtgccaatgtcaaaagccagcagatgctgtgagatccagatctgactctc  
 aggaaggtctcttact□

5 Actual comp position 89543 sequence gtgaaaccataaaccaagc  
 Reverse comp position 89543 sequence gcttgggtttatgggtttcac  
 g c t a toalno totalvalue 3 7 2 9 21 62

10 Actual comp position 90103 sequence ctccatgtctctgaacatcag  
 Reverse comp position 90103 sequence ctgatgttcagagacatggag  
 g c t a toalno totalvalue 3 7 6 5 21 62

=====

15

An additional rule relating to gene family members may also be included in the set of primer selection rules. Many genes in the human genome are members of gene families, which means that they closely resemble other genes at other positions in the genome. When primer sequences are selected for a certain gene, one may later find that the selected primers are actually undesirably present in these other family members. The cycle of selecting an appropriate primer sequence for a given gene, that is, identifying a candidate primer sequence, searching the public database to find out whether or not it is specific to that gene, identifying that it is not specific to the gene, reselecting another candidate primer sequence, etc., could go on for several loops before an appropriate primer sequence is identified.

An example command for operating the function for this task is:

```
primer611 sult1a1.txt sult1a1join.txt primerout sult1a2.txt sult1a3.txt
```

30 where the program executable command is primer611, the input sequence file within which to find primers is sult1a1.txt, the input join file that tells the program where the



coding (exons) regions is sult1a1join.txt , the output file is primerout, and the other two files, sult1a2.txt and sult1a3.txt, are sequence files of family members. The number of gene family files which may be included can be large.

When the program selects a candidate primer in the sult1a1.txt file, it then reads the sult1a2.txt and sult1a3.txt files to see if it is present. If it is present, it discards it and selects another candidate primer. If it is not present in the files, it selects and stores it and goes on to find the next primer. The program also looks at the family member files in both forward and reverse directions to be complete and eliminate the user from having to format these files to be in the proper coding orientation.

Thus, the software can select primers that are unique to the gene of interest and can be relied upon for genes that are members of families. This functionality can be added to the functionality of picking the best primers around the exons of a gene for the primer design process -- select the candidate primer only if it is unique to the target file and not present in the gene family files.

To further illustrate the functionality and output, below is a listing of the primeronly file and a portion of the primerout file (listing the 1<sup>st</sup> three primer pairs). The command used to generate this output is:

```
primer611 topo2a.txt topo2ajoin.txt primerout topo2b.txt chr18.txt.
```

The primerout file is defined in the fourth element of the above command and the primeronly file below is created and named automatically. The primerout file has

each of the exon regions defined in the topo2ajoin.txt file printed out with "....." before and after the exon, and documents the steps that the program went through when picking the primers. The primerout file lists candidate primer sequences that otherwise met the primer selection rules, but was found in one of the gene family files and was therefore rejected (see areas that read "FOUND in"). The output presentation allows a user to go back to a specific region and redesign a primer if the primer selected happens to be in a repetitive sequence region not screened out with the gene family files. This may be done, for example, by doing a database search.

```

=====
"PRIMERONLY" FILE
=====

topE1E2-5    actgtggaacagccagtaga
□
topE1E2-3    tcttgataacctcgctgtgtc
□
□
topE3E4E5-5
□
topE3E4E5-3
□
□
topE6E7E8-5  atgtgccaccctctatccag
topE6E7E8-3  ttagagatgatgaataaagctcc

topE9E10E11-5  cccagcctaacagttcttttg
topE9E10E11-3  ccactacgctcggccaattt

topE12E13E14-5  aagagaacagtaactcccgtc
topE12E13E14-3  cagcactgattccatgcatac

topE15-5      gccagaagttgtaggttcaag
topE15-3      cttactcagtcccaagctct

topE16-5      gcgtgacacatagcaagtgc
topE16-3      gccagttcttcaatagtaccc

topE17E18E19-5  gagaagaacctttgccaatgg
topE17E18E19-3  ctccaccattactctcaccaa

topE20E21E22-5  tgcctgtataccgggatatac

```

**0201-0001-Frudakis**

	21	14332	14496	topE21
	22	14628	14711	topE22
	23	16803	16934	topE23
	24	18702	18854	topE24
5	25	19098	19221	topE25
	26	19328	19371	topE26
	27	19799	19933	topE27
	28	21275	21474	topE28
	29	21792	22080	topE29

10

SORTED JOIN Values .....

	1	1	66	topE1
	2	290	502	topE2
15	3	1443	1616	topE3
	4	1806	1907	topE4
	5	2015	2152	topE5
	6	4630	4768	topE6
	7	5136	5293	topE7
20	8	5586	5711	topE8
	9	6318	6428	topE9
	10	6571	6676	topE10
	11	6767	6876	topE11
	12	8378	8470	topE12
25	13	8770	8884	topE13
	14	8988	9109	topE14
	15	10207	10355	topE15
	16	12180	12411	topE16
	17	12598	12732	topE17
30	18	12852	13052	topE18
	19	13194	13389	topE19
	20	14138	14229	topE20
	21	14332	14496	topE21
	22	14628	14711	topE22
35	23	16803	16934	topE23
	24	18702	18854	topE24
	25	19098	19221	topE25
	26	19328	19371	topE26
	27	19799	19933	topE27
40	28	21275	21474	topE28
	29	21792	22080	topE29

COMBINED JOIN Values .....

45

	1	1	502	topE1E2
	2	1443	2152	topE3E4E5
	3	4630	5711	topE6E7E8
	4	6318	6876	topE9E10E11
50	5	8378	9109	topE12E13E14
	6	10207	10355	topE15
	7	12180	12411	topE16
	8	12598	13389	topE17E18E19
	9	14138	14711	topE20E21E22
55	10	16803	16934	topE23
	11	18702	18854	topE24
	12	19098	19933	topE25E26E27

Total no of joins : 13

5

```

PAIR NO :      1           First      1           Second    502           Name
topE1E2
PAIR Length .....           501

Block Length .....           :    1301

Block starting position..... :      0

```

15

[illegible]

Did not get PRIMER , what to do , DO NOT HAVE ENOUGH CHARACTERS: 1 TO  
DEAL

Seq ..	tcttgataacctcgctgtgtgc	FOUND	in	:	chr18.txt	at	8964	position
Seq ..	ttgataacctcgctgtgtcac	FOUND	in	:	chr18.txt	at	8966	position
Seq ..	gataacctcgctgtgtcacc	FOUND	in	:	chr18.txt	at	8968	position
Seq ..	ataacctcgctgtgtcaccc	FOUND	in	:	chr18.txt	at	8969	position
Seq ..	caggctggagtacagtgatg	FOUND	in	:	chr18.txt	at	8988	position
Seq ..	aggctggagtacagtgatgc	FOUND	in	:	chr18.txt	at	8989	position
Seq ..	ctggagtacagtgatgcaatc	FOUND	in	:	chr18.txt	at	8992	position
Seq ..	ggagtacagtgatgcaatcac	FOUND	in	:	chr18.txt	at	8994	position
Seq ..	gagtacagtgatgcaatcacg	FOUND	in	:	chr18.txt	at	8995	position
Seq ..	agtacagtgatgcaatcacgg	FOUND	in	:	chr18.txt	at	8996	position
Seq ..	cagtgatgcaatcacggctc	FOUND	in	:	chr18.txt	at	9000	position
Seq ..	gtgatgcaatcacggctcac	FOUND	in	:	chr18.txt	at	9002	position
Seq ..	gcaatcacggctcactacag	FOUND	in	:	chr18.txt	at	9007	position
Seq ..	caatcacggctcactacagc	FOUND	in	:	chr18.txt	at	9008	position
Seq ..	aatcacggctcactacagcc	FOUND	in	:	chr18.txt	at	9009	position
Seq ..	tcaagcgatcatccacctc	FOUND	in	:	chr18.txt	at	9043	position
Seq ..	aagcgatcatccacctcag	FOUND	in	:	chr18.txt	at	9045	position
Seq ..	gatcatccacctcagcttc	FOUND	in	:	chr18.txt	at	9049	position
Seq ..	tcatccacctcagcttctg	FOUND	in	:	chr18.txt	at	9051	position
Seq ..	cacctcagcttctggagtag	FOUND	in	:	chr18.txt	at	9057	position
Seq ..	acctcagcttctggagtagc	FOUND	in	:	chr18.txt	at	9058	position
Seq ..	ctcagcttctggagtagctg	FOUND	in	:	chr18.txt	at	9060	position



ttcactctcatttgaaatctttcaatgacttccactcagggaaagtccaaattccataatttgccaacaagaaga  
tctgctgtaattctaattacacactacttctccaactcatctcagtgccagtttttcgtatattgtcctgttgctttta  
aattactgaaaagcacagtgtctttccccSeq .. ccattcccttatgcctatcag FOUND in :  
chr18.txt at 9221 position

5 Seq .. gaccattcccttatgcctatc FOUND in : chr18.txt at 9219 position  
Seq .. tcaagtgatccacccacctc FOUND in : chr18.txt at 9182 position  
Seq .. actcctgggctcaagtgatc FOUND in : chr18.txt at 9172 position  
Seq .. tgaactcctgggctcaagtg FOUND in : chr18.txt at 9169 position  
Seq .. cttgaactcctgggctcaag FOUND in : chr18.txt at 9167 position  
10 Seq .. aggctggtcttgaactcctg FOUND in : topo2b.txt at 36055 position

PRIMER 1: 1246 ... tcaactatgttgcccaggctg

Letters	20 g count	5 t count	6 c count	6 a count	3 total
15 62	topE3E4E5-5 tcaactatgttgcccaggctg				
Seq ..	gcctaagacttgctttcagtc	FOUND	in :	chr18.txt at 10319	position
Seq ..	cctccataactcactgatttgc	FOUND	in :	chr18.txt at 10365	position
Seq ..	ctccataactcactgatttgc	FOUND	in :	chr18.txt at 10366	position
20 Seq ..	tccataactcactgatttgc	FOUND	in :	chr18.txt at 10367	position
Seq ..	cactgatttgcatacaagc	FOUND	in :	chr18.txt at 10375	position
Seq ..	ctgatttgcatacaagcag	FOUND	in :	chr18.txt at 10377	position
Seq ..	tgatttgcatacaagcagc	FOUND	in :	chr18.txt at 10378	position
Seq ..	tttgccatacaagcagccc	FOUND	in :	chr18.txt at 10381	position
25 Seq ..	cccaaccaacctctaggttg	FOUND	in :	chr18.txt at 10445	position
Seq ..	taaacaagaaagctgggagcc	FOUND	in :	chr18.txt at 10467	position
Seq ..	caagaaagctgggagccttc	FOUND	in :	chr18.txt at 10471	position
Seq ..	aagaaagctgggagccttcc	FOUND	in :	chr18.txt at 10472	position
Seq ..	ctgggagccttcctttatttc	FOUND	in :	chr18.txt at 10479	position
30 Seq ..	tgggagccttcctttatttc	FOUND	in :	chr18.txt at 10480	position
Seq ..	gaatcatctcttgatgctgc	FOUND	in :	chr18.txt at 10525	position
Seq ..	atcatctcttgatgctgcag	FOUND	in :	chr18.txt at 10527	position
Seq ..	atctcttgatgctgcagtag	FOUND	in :	chr18.txt at 10530	position
Seq ..	ctcttgatgctgcagtagc	FOUND	in :	chr18.txt at 10532	position
35 Seq ..	ggatgctgcagtagcttctc	FOUND	in :	chr18.txt at 10537	position
Seq ..	tgctgcagtagcttctcacc	FOUND	in :	chr18.txt at 10540	position
Seq ..	ctggttaagtctttccttgg	FOUND	in :	chr18.txt at 10605	position
Seq ..	ttcaatgacttccactcaggg	FOUND	in :	chr18.txt at 10689	position
Seq ..	atgacttccactcagggaaag	FOUND	in :	chr18.txt at 10693	position
40 Seq ..	cttccactcagggaaagtcc	FOUND	in :	chr18.txt at 10697	position
Seq ..	ctcagggaaagtccaaattcc	FOUND	in :	chr18.txt at 10703	position
Seq ..	tggccaacaagaaagatctgc	FOUND	in :	chr18.txt at 10730	position
Seq ..	gccaacaagaaagatctgctg	FOUND	in :	chr18.txt at 10732	position
Seq ..	cactacttctccaactcatc	FOUND	in :	chr18.txt at 10764	position
45 Seq ..	cctacttctccaactcatctc	FOUND	in :	chr18.txt at 10766	position
Seq ..	cttctccaactcatctcagtg	FOUND	in :	chr18.txt at 10770	position
Seq ..	ttctccaactcatctcagtg	FOUND	in :	chr18.txt at 10771	position
Seq ..	ctccaactcatctcagtgcc	FOUND	in :	chr18.txt at 10773	position
Seq ..	ccaactcatctcagtgccag	FOUND	in :	chr18.txt at 10775	position

Did not get PRIMER , what to do , DO NOT HAVE ENOUGH CHARACTERS: 2208 TO DEAL

55 PAIR NO : 3 First 4630 Second 5711 Name  
topE6E7E8  
PAIR Length ..... 1081

```
Block starting position..... : 3930
```

25  
30  
35

Letters	20 g count	3 t count	5 c count	8 a count	4 total
a					
b					
c					
d					
e					
f					
g					
h					
i					
j					
k					
l					
m					
n					
o					
p					
q					
r					
s					
t					
u					
v					
w					
x					
y					
z					
total					

PRIMER 2 actual : 6005 ... gagtgcaatgggtgcaatcttg

```
reverse          : 6005 ... caagattgcaccattgcactc
topE6E7E8-3     caagattgcaccattgcactc
```

**0201-0001-Frudakis**



\*  
\*  
\*

5

=====

There are two gene family files in this comparison. The topo2b.txt file is a  
10 human genome sequence for a gene called topoisomerase 2b, which is highly related to  
the gene of interest, topoisomerase 2a. In the primerout file, many of the candidate  
primers the program selected were present in this family member and were therefore  
rejected. This demonstrates the utility of the functionality of this program. The second  
family member sits on chromosome 18 and is a pseudogene (a duplicated region of  
15 DNA that does not make a real gene -- a serious nuisance for designing primers that are  
to amplify a single genetic position). The program has accommodated for this as well; it  
selected a candidate primer that was found in this file a large number of times.

Without this functionality, primers that would amplify three different regions at  
the same time would be designed: the topo2a region of interest; the topo2b region  
20 related to it; and a nuisance region in chromosome 18. Unfortunately, the resulting data  
would show numerous discrepancies that are not real polymorphisms. These  
sequences are actually from different genetic positions that are highly similar to one  
another but not identical. Thus, most of the "SNPs" found in this manner are not SNPs  
at all. If one tried to genotype people at a "false SNP," they would get incoherent data  
25 as they would be looking at three different positions within the genome at the same

time. It is important to produce data for single positions at a time so that the data can be accurately read and interpreted.

Advantageously, the rules that the inventive software uses in the preamplification process are different than those of conventional programs in that they are suitable for use in designing high throughput experiments where many different things can be done simultaneously. It is more efficient to do simultaneous amplifications of four or five regions in 500 people, for example, rather than doing them one by one. This is where the rule regarding the fixed predetermined annealing temperature (e.g., 62° Celsius) comes into play: since all of the primers selected by the program have the same annealing temperature, the work can be done more efficiently. Another example is where the software automatically decides if a single primer pair can be utilized for two or more coding regions, which saves additional time and expense. Furthermore, the rule regarding gene family data is important for generating reliable output data and for efficiency.

The output of the software is also unique. The numbers included in the output use the numbering pattern that exists in the input sequence file (for example, starting at "10003") rather than starting at "1" like most other programs. This means that a primer at position "11234" can be quickly located, whereas in other programs the number for the primer would be "1231" and one would have to perform the math to figure out its location. This is particularly important for those primers that have to be redesigned manually due to having certain characteristics that can only be determined through a database search.

### Additional Details Regarding The Discovery of Reliable SNP and Haplotype

Data. The description that follows provides additional details regarding steps 318-342 of FIG. 3B, which may be referred to as part of the post-amplification process. As described earlier, one important goal of the program is to find reliable discrepancies between individuals at a sequence of a particular genetic locus or location in the genome. To do this, the inventive methods use a direct measure of the nucleotide base quality, or "phred" score, of an observed discrepancy (at steps 326-328 of FIG. 3B).

Actual DNA sequence data files, called chromatograms, are utilized as input, as quality information is an inherent part of such files. As is well-known, a sequence chromatogram looks like a series of colorful peaks and valleys. The color of a peak indicates the DNA base present at that position in the sequence. Peaks in a graph for a good sequence tend to be higher than for a bad sequence, and overlapping peaks tend to indicate poor reliability. Such information is used to determine whether a discrepancy in a sequence alignment represents a good candidate SNP or not.

The functionality of a conventional phred program is used to call the quality of every letter, and the program aligns the sequences and finds where they are "reliably" different from one another. By reliable, it is meant that the differences in sequence are differences between letters of good quality. An example of one such program is the phred program available from the University of Washington, which ascribes a numerical value to indicate the quality of each letter of a sequence. The phred functionality makes a separate file with all of these numbers, for each letter.

DNA sequences from various individuals are aligned using a conventional sequence alignment algorithm (at step 320), such as that provided using conventional Clustal software functions available by and from the EMBL, Heidelberg Germany, and is a re-write of the popular Clustal V program described by Higgins, Bleasby, and Fuchs (1991) CABIOS, 8, 189-191 (Thompson, J.D., Higgins, D.G. and Gibson, T.J. (1994) (CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, positions-specific gap penalties and weight matrix choice. Nucleic Acids Research, 22:4673-4680). Thus, the sequence alignment file is the first input file to the program. Any discrepancy that occurs within a neighborhood of other discrepancies is recognized so that the quality value information can be checked. If this information is greater than predetermined quality information, such as a user-defined input value, it is accepted and presented to the user for final acceptance. If not, it is discarded. The quality control file created from the phred functionality serves as the second input file.

In the sequence within which the discrepancy occurs, positions of the minor letters of the discrepancy are presented to the end-user. This lets the end-user contemporaneously call up the raw DNA sequence chromatogram and find the actual trace data peak for the letter. This is advantageous because a visual inspection of raw DNA sequence data is the most reliable method of determining whether a discrepancy is valid. While the purpose of the software is to eliminate many time consuming steps, in some cases, borderline quality values nonetheless necessitate its execution. The presentation of the precise position and relevant file names for a discrepancy makes this

step easy to execute. Also, the end-user is shown presentations of discrepancies that do not meet the quality control criteria. This is important because, in some cases, a borderline quality value may conceal good data due to other problems with sequence compressions or peak spacing.

5 Another important attribute is afforded the software because it can recognize reliable base deletion polymorphisms. This is performed by parsing the phred quality data for the bases surrounding the deletion in randomly selected sequences which contain the deletion. With conventional programs, if a discrepancy is a deleted base there is no quality control information to check since no data is produced for a non-base (and there is consequently no phred value for the deleted base). This eliminates any  
10 discovery of single base deletion polymorphisms. Deletion polymorphisms are common and, since the goal is to thoroughly document the various genetic haplotypes in a population, a SNP-finding program that can recognize deletion polymorphisms offers competitive advantages. Not knowing all of the variants in a gene sequence causes the resolution of haplotype-based studies to be sub-optimal, compared to being  
15 able to recognize all variants (including deletion polymorphisms).

The software may also incorporate rules to maximize efficiency during these steps. For example, the program may focus on determining the phred value for discrepancies that fall within a block of sequence with an acceptable average phred  
20 value. As another example, the user-defined phred value could be different for different regions of the sequence. In another variation, the program is configured to recognize amino acid differences by translating the sequences and instructed to only

present candidate polymorphisms that result in a change in amino acid sequence.

Example Walk-Through. Input = (1) Clustal W alignment file and (2) phred quality file. The user inputs a minor letter phred quality control value for the current run, as well as a local phred quality control value. For example, the user may enter the values "24" and "17" for the the minor letter and local phred quality control values, respectively. Then, from the first input file, each column (position or slice) of the alignment is analyzed to determine whether the column is homogeneous (i.e., whether each sequence has the same letter at that position) or heterogeneous (i.e. whether there are two or more different letters at that position).

As an example, consider the following:

AHRE11-3	AGGGGGTAGATTTTAAAAAT-CATGTTAATGTTATTTACT-
AHRE11-3-E10	AGGGGGTAGATTTTAAAAAT-CATGTTAATGTTATTTACT-
AHRE11-3a	AGGTGTAAGATTTTAAAAATACATGTTAATGTTATTTACT-
AHRE11-3u	AGGGGTA-GATTTCAAAAATACATGTTAATGTTATTTACT-
14	AGGGGTA-GATTTTAAAAATACATGTTAATGTTATTTACT-
AHRE11-3-C4	AGGGGTAAGATTTTAAAAATACATGTTAATGTTATTTACT-
AHRE11-3-D5	AGGGGTAAGATTTTAAAAATACATGTTAATGTTATTTACT-

The first column of letters is homogeneous. So is the second and third. The fourth is heterogeneous, as is the sixth, etc.

The second input file is the phred quality file, which takes the format of the 1XN matrix below for each sequence. The entry for the first sequence above (AHRE11-3) appears below:

```
>AHRE11-3 folder=AHRE11-3 length=414
8 9 23 24 32 34 27 27 34 34 32 32 34 34 32 32 29 29 26 26 26 28 34 31 29 29
32 35 35 35 45 45 45 40 35 35 39 32 33 32
```

In this file, the first two letters are of very low quality or reliability because, for biochemical reasons, sequencing reactions routinely have trouble at the beginning of a sequence read.

For each column of the alignment, the software recognize whether there is a  
5 discrepancy (i.e., major and minor letters.) If a discrepancy exists, then the following logic is executed:

For each minor letter, read the phred value.  
For example, in column 14 above, sequence AHRE11-3u has a C but the others have a T. The "C" is a minor letter and it has the value 34.

Calculate the average phred value for the major letter (G in column 14 above)

Calculate the average phred value for each minor letter (in column 14 above, there is only one minor so this is the same as the phred value for that letter.

Determine the number of major letters.

Determine the number of minor letters.

Calculate the average phred value for the block of letters 7 in front and 7 behind the column using all of the input sequences and their quality values. This will be called the local phred quality value.

To process the job, the phred value of the minor letter and average phred value of the  
25 major letter are utilized such that

If the phred value of any minor letter in the column is greater than the user-defined threshold value,

And

If the average phred value of the major letter for the column is above a different threshold value defined by the user,

Then label the column as accepted and present to the user for visual inspection.

Alternatively, a more sophisticated method for determining the worth of a positional column is to use a function to calculate the probability that a column contains

a reliable polymorphism using the average quality value for the column, the quality values for the minor letters, the quality value for the region around the column (using all the sequences), or other variables. For this approach the following logic is utilized:

- 1) A column with a high average major letter phred score and a high minor letter phred score is a better column than one with
  - a) a low average major letter phred score and a high minor letter phred score;
  - b) a high average major letter phred score and a low minor letter phred score;
  - c) a low average major letter phred score and a low minor letter phred score; and
- 2) A column with a discrepancy in a region of sequence that has a high local phred quality value is better than one in a region with a low local phred quality value.

Preferably, a probability function is employed for this task, including variables for that which is measured above. For example, one might use Bayes' theorem to calculate this probability; for every column a vector is created from the variables calculated above and the linear equation:

$$y = A_1X_1 + A_2X_2 + A_3X_3 \dots A_nX_n$$
  
giving the vector  $Y = (A_1, A_2, A_3 \dots A_n)$ , where  $A_n$  are parameters.  
Then determine a Bayesian estimate  
 $p(w|x) = [p(x|w)p(w)]$  divided by  $p(x)$ ,  
where  $p(w|x)$  = classification score of the column as good or bad or somewhere in between (called the posterior probability),  $p(x)$  is the frequency or uniqueness or worth of this vector, and  $p(w)$  is the frequency or uniqueness of the class.  $P(x|w)$  is the conditional probability that  $x$  is observed given that  $w$  is also observed - in this frequency that vectors of the above  $A_n$  are observed for true SNP columns (determined using other suitable biochemical techniques).



Once the alignment file has been inspected for every column, the results are presented to the user. For example, if the probability is high that a column contains a reliable polymorphism, then the column is presented to the user along with 7 letters in front and 7 letters behind for each sequence in the alignment. For example,

5  
Sequence 1 TTTATCTGACTGGAG  
Sequence 2 TTTATCTGACTGGAG  
Sequence 3 TTTATCTCACTGGAG

10 Also, the "average" sequence 200 letters in front and 200 letters behind the column is presented. For example,

15  
ATTATGCTCG ATTATGCTCG ATTATGCTCG ATTATGCTCG ATTATGCTCG  
ATTATGCTCG ATTATGCTCG ATTATGCTCG ATTATGCTCG ATTATGCTCG  
ATTATGCTCG ATTATGCTCG ATTATGCTCG ATTATGCTCG ATTATGCTCG  
ATTATGCTCG ATTATGCTCG ATTATGCTCG ATTATGCTCG ATTATGCTCG G/C  
ATTATGCTCG ATTATGCTCG ATTATGCTCG ATTATGCTCG ATTATGCTCG  
ATTATGCTCG ATTATGCTCG ATTATGCTCG ATTATGCTCG ATTATGCTCG  
20 ATTATGCTCG ATTATGCTCG ATTATGCTCG ATTATGCTCG ATTATGCTCG

In the above example, there is only one column with discrepancies; each of the other columns are homogeneous. In practice, this will be unusual and the presentation will

25 look more like the following (note the letters R, Y, M):

30  
YTTATGCTCG ATTATGCTCG ATTATGCTCG ATTATGCTCG ATTATGCTCG  
ATTATGCTCG ATTATGCTCG RTTATGCTCG ATTATGCTCG ATTATGCTCG  
ATTATGCTCG ATTATGCTCG ATTATGCTCG ATTATGCTCG ATTATGCTCG  
ATTATGCTCG ATTATGCTCG ATTATGCTCG ATTATGCTCG ATTATGCTCG S  
ATTATGCTCG ATMATGCTCG ATTATGCTCG ATTATGCTCG ATTATGCTCG  
ATTATGCTCG ATTATGCTCG ATTATGCTCG ATTATGCTCG ATTATGCTCG  
ATTATGCTCG ATTATGCTCG ATTATGCTCG ATTATGCTCG ATTATGCTCG  
35 ATTATGCTCG ATTATGCTCG ATTATGCTCG ATTATGCTCG ATTATGCTCG

Where

R=A or G

Y=C or T  
 K=G or T  
 M=A or C  
 S=G or C  
 5 W=A or T  
 N=any base  
 B=C,G, or T  
 D=A,G or T  
 H=A,C or T  
 10 V=A,C or G

Other information may also be presented, such as the following: (a) for each sequence with a minor letter, the sequence name and the associated phred value for the minor letter; and (b) the local region phred score.

Example Output. Below is a file that shows what the software produces as it inspects a single discrepancy.

```

=====
k = 70
Position of Reference sequence without dashes :
65
Position of complement sequence: 209

Indicator      ^

QUALITY INFORMATION
  Discrepancies at position
70
Minor letter 1::-:1
Minor letter 2::A::1
Major letter ::G::60
  Got '-' as minor value

  Got 1
minor characters
  Minor characters ::: A

  Check quality for mlnor A

Got sequence , sequence
id AHRE9-5-D7
No of dashes before minor
character position      67
Quality value (
4) is lessthan24 at position 4
Total No of minor charaters quality is less than24 is 1
  
```

Total No of minor charaters  
quality is greater than 24 is 0

	AHRE9-5-D2	C-TCTGAGTTA;Accumulated	SNP # : 0 S
5	AHRE9-5-H1	C-TCTGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-C4	C-TTTGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-B5	C-TCTGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-D5	C-TTTGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-A6	C-TCTGAGTTA;Accumulated	SNP # : 0 S
10	AHRE9-5-B2	C-TCTGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-C3	C-TCTGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-C2	C-TCTGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-D3	C-TCTGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-E2	C-TTTGAGTTA;Accumulated	SNP # : 0 S
15	AHRE9-5-F2	C-TCTGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-E1	C-TCTGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-G2	C-TCTGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-G3	C-TCTGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-H2	C-TTTGAGTTA;Accumulated	SNP # : 0 S
20	AHRE9-5-D1	C-TTTGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-F1	C-TTTGAGTTA;Accumulated	SNP # : Q S
	AHRE9-5-D12	CATTCGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-B4	CAT-CGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-D6	CAT-CGAGTTA;Accumulated	SNP # : 0 S
25	AHRE9-5-C1	CAT-CGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-A12	CAT-CGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-B11	CAT-AGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-D7	--AATAGAGTA;Accumulated	SNP # : 1 S
	AHRE9-5-H12	-----GGTTA;Accumulated	SNP # : 0 S
30	AHRE9-5-D4	C-TCTGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-C5	C-TCTGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-B1	C-TCTGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-B3	C-TCTGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-A3	C-TCTGAGTTA;Accumulated	SNP # : 0 S
35	AHRE9-5-C6	CAT-CGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-F11	C-TCCGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-G11	C-TCCGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-C12	C-TTCGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-E10	C-TCCGAGTTA;Accumulated	SNP # : 0 S
40	AHRE9-5-C10	CTC-CGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-G12	CTCNCGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-D10	CATTCGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-D8	CATTCGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-D9	CATCCGAGTTA;Accumulated	SNP # : 0 S
45	AHRE9-5-E11	C-TCCGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-C9	CAT-TGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-E8	TATTCGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-B10	TCATCGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-D11	TCTTCGAGTTA;Accumulated	SNP # : 0 S
50	AHRE9-5-C8	CAT-CGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-B8	TCTTCGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-F8	TCTCNGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-H11	TCTCCGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-A8	CAT-CGAGTTA;Accumulated	SNP # : 0 S
55	AHRE9-5-F12	C-TTCGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-E12	C-TCCGAGTTA;Accumulated	SNP # : 0 S
	AHRE9-5-F7	CATCCGAGTTA;Accumulated	SNP # : 0 S

```

AHRE9-5-G10 C-TCCGAGTTA;Accumulated SNP # : 0 S
AHRE9-5-B9 C-TTCGAGTTA;Accumulated SNP # : 0 S
AHRE9-5-C7 --CTTGAGT-A;Accumulated SNP # : 0 S
AHRE9-5-F10 AATCCGAGTTA;Accumulated SNP # : 0 S
5 AHRE9-5-C11 CATTCCGAGTTA;Accumulated SNP # : 0 S
AHRE9-5-A10 ACTCCGAGTTA;Accumulated SNP # : 0 S
AHRE9-5-F9 C-TCCGAGTTA;Accumulated SNP # : 0 S
AHRE9-5-G8 C-TCCGAGTTA;Accumulated SNP # : 0 S

10 Left :
Right :
AGTTACAATGATATAATCTGGTCTTCCATTTTTATAAAGCAGGCGTGCATTAGACTGGACCCAAGTCCATCG
GTTGTTTTTTGTAAGAAGCCGGA-
AAACTATCATGCCACTTCTCCANTCTTAATCACTAAAATAAAATTAATAWA---
15 ATTAAATTATCAAACCCCAATC-AATATAGTAAAGATTATTCCTAAAA

```

```

Do you want to choose this into SNP data ?[y/n] n
*****

```

```

=====

```

Now consider the text window below which shows an alignment produced by the software. Note the small numbers at the end of most of the lines (most are 0, some 1; one 17, one 22). When a discrepancy in the last two sequences having a quality score on the borderline is seen, and the number of "Accumulated SNPs" is high as it is shown in the last two lines, the discrepancy can be ignored as the large number indicates that the sequence is of poor quality. This inference is good because real SNPs occur at a frequency of about 1 in 200 letters and the high numbers are much greater than one would expect. If it were not for these numbers, one would have to go and look at the sequence trace file to see if the discrepancy was real or not. Using this technique, it has never been observed that a discrepancy in a sequence with a large Accumulated SNP number turns out to be a real SNP upon visual inspection of the trace data. Thus, time can be saved by avoiding to have to regularly view such trace data.

```

=====
S13462.DPG-51-CP1 ACAATCCTTAA;Accumulated SNP # : 0 S
S13462.DPG-90-CP1 ACAATCCTTAA;Accumulated SNP # : 0 S

```



Note that the second sequence is "shifted" relative to the other three due to one single sequencing mistake called an insertion, which is common. The alignment program is not perfect and does not always make the correct alignment by shifting the sequences relative to one another. Even though the quality values for the letters A, T, A, A, T and T are very good, they are not SNPs but rather sequencing/alignment errors. Most other programs would output these letters as good candidate SNPs, so if the end-user did not go back to the data to inspect it valuable time and expense would be incurred by designing genotyping experiments based on incorrect data.

The inventive program avoids this by visually presenting a local neighborhood of sequences to the end-user for those discrepancies that meet the phred threshold value. In other words, the program presents a block of sequences (such as the one above) so that an experienced user can recognize common errors such as this shift error.

Other common errors the end-user might notice are discrepancies in strings of sequence (such as GGGGG), or a phenomena called "bleedthrough". A conventional program relying just on phred score would select those mistakes and bad experiments would subsequently be designed. Since the inventive program shows the local sequence around this region for all the sequences, it is obvious to a trained molecular biologist that the finding by the software is incorrect and should be discarded.

So one advantage of the software is that it presents a snapshot of the data, along with a query line asking if the user wishes to accept the data or not, so that invaluable human input is included in the SNP discovery analysis.

2) Another advantage is that the precise position and sequence that the discrepancy occurs is readily apparent to the user. The example output above shows how this data is presented. Notice that each discrepancy is advantageously identified by using `k = "column number"`. This is important in case the end-user wants to call up the sequence data electropherogram, since it tells him which one to call up and where to go to see the relevant base. This is often done in different windows on the desktop. Visual inspection of raw DNA sequence data is the most reliable method of determining whether a discrepancy is valid. While the purpose of software is to eliminate such time consuming steps, in some cases borderline quality values require visual inspection. The presentation of the precise position and relevant file names for a discrepancy makes this step easy to perform.

3) Another advantage is that the end-user can specify a quality control value for a run of the program, then go back and repeat the run using a different quality control value. The quality for a position that meets the threshold requirements is also reported to the user so that borderline cases can be further reviewed.

4) Yet even another advantage is that the program presents the neighboring 200 letters of average sequence (for all of the individuals in an analysis) in front of and behind candidate SNP locations. This is important because when submitting SNP locations to a SNP consumables company (e.g., Orchid), one must submit the neighboring sequence as well so that the kit can be designed to assay this SNP in thousands of people.

5) Finally, another advantage is that the user can visualize deletion mutations, which do not have corresponding phred values. A unique attribute is afforded the software because of this functionality. The program can recognize reliable base deletion polymorphisms and present them to the user for visual inspection. In conventional programs, if a discrepancy is a deleted base there is no quality control information to check since no data is produced for a non-base or deleted base (and there is consequently no phred value for the deleted base). This would eliminate the discovery of single base deletion polymorphisms. Deletion polymorphisms are common and, since the goal is to thoroughly document the various genetic haplotypes in a population, a SNP finding program that can recognize deletion polymorphisms offers competitive advantages. Not knowing all of the variants in a gene sequence causes the resolution of haplotype-based studies to be sub-optimal, compared to being able to recognize all of the variants.

In an alternate embodiment, the software does not use actual DNA sequence data files or chromatograms but rather accepts and utilizes sequence information in text format which is freely available and downloadable from publicly available databases. For quality control, an indirect measure of quality is used. For example, any discrepancy that occurs within a bleedthrough region, or within the neighborhood of discrepancy clusters is ignored.

It should be readily apparent and understood that the foregoing description is only illustrative of the invention and in particular provides preferred embodiments thereof. Various alternatives and modifications can be devised by those skilled in the



art without departing from the true spirit and scope of the invention. For example, gene data from human, animal, plant, or other may be utilized in connection with the methods. Accordingly, the present invention is intended to embrace all such alternatives, modifications, and variations which fall within the scope of the appended  
5 claims.

What is claimed is:

0964959.02604  
F09260"65079660